BÁO CÁO KỸ THUẬT

HỆ THỐNG PHÁT HIỆN ẢO GIÁC TRONG MÔ HÌNH NGÔN NGỮ TIẾNG VIỆT

1. TỔNG QUAN BÀI TOÁN

Phát hiện và phân loại hallucination (ảo giác) trong câu trả lời của mô hình ngôn ngữ tiếng Việt thành 3 loại:

- `no`: Câu trả lời hoàn toàn được hỗ trợ bởi ngữ cảnh, không có thông tin bịa đặt
- `intrinsic`: Câu trả lời mâu thuẫn hoặc xuyên tạc thông tin từ ngữ cảnh
- `extrinsic`: Câu trả lời thêm thông tin mới không có trong ngữ cảnh

2. PHƯƠNG PHÁP TIẾP CẬN

2.1. Kết Hợp Few-Shot Learning và LoRA Fine-tuning

Hệ thống áp dụng kết hợp hai phương pháp:

Few-Shot In-Context Learning:

- Mỗi mẫu dữ liệu được cung cấp 5 ví dụ mẫu trong prompt
- Đảm bảo có ít nhất 1 ví dụ cho mỗi loại label (no, intrinsic, extrinsic)
- Các ví dụ được chọn ngẫu nhiên nhưng deterministic (seed theo ID sample)
- Giúp mô hình học cách phân loại thông qua các ví dụ cụ thể

LoRA Fine-tuning:

- Tinh chỉnh mô hình trên dữ liệu huấn luyện để học pattern phân loại
- Chỉ train các adapter weights thay vì toàn bộ mô hình
- Giảm đáng kể tài nguyên cần thiết (VRAM, thời gian training)

2.2. Cấu Trúc Prompt

Prompt được thiết kế với 3 phần chính:

1. **INSTRUCTION**: Định nghĩa rõ ràng task và 3 loại label, kèm quy tắc phân loại chi tiết
2. **EXAMPLE CLASSIFICATION**: 5 ví dụ mẫu với đầy đủ Context, Prompt, Response, Label, và Explanation
3. **QUERY**: Mẫu cần phân loại với Context, Prompt, Response và yêu cầu đưa ra Label

2.3. Quy Tắc Phân Loại

Thứ tự ưu tiên đánh giá:

1. Kiểm tra mâu thuẫn với context → intrinsic
2. Kiểm tra thông tin thêm vào không có căn cứ → extrinsic
3. Nếu hoàn toàn được hỗ trợ → no

Nguyên tắc:

- Mâu thuẫn được ưu tiên hơn thông tin thêm vào
- Đánh giá ở mức semantic, bỏ qua lỗi chính tả nhỏ
- Nếu chỉ nói "không đủ thông tin" mà không bịa đặt → no

3. MÔ HÌNH VÀ KỸ THUẬT

3.1. Base Model

Qwen3-4B-Instruct-2507

- Kích thước: 4 tỷ tham số
- Loại: Instruction-tuned model đa ngôn ngữ (hỗ trợ tiếng Việt)
- Framework: Unsloth (tối ưu hóa cho training nhanh)

3.2. Kỹ Thuật LoRA (Low-Rank Adaptation)

Cấu hình:

- Rank (r): 32
- Alpha: 32
- Dropout: 0
- Target modules: Tất cả attention layers (Q, K, V, O) và FFN layers (gate, up, down)

Lợi ích:

- Giảm VRAM từ ~80GB xuống ~24GB
- Chỉ train < 1% tham số của mô hình
- Tốc độ training nhanh hơn full fine-tuning 3-5 lần
- Dễ dàng lưu trữ và deploy (chỉ cần lưu adapter weights)

3.3. Quantization

4-bit Quantization:

- Giảm kích thước model từ 16GB xuống ~4GB
- Duy trì chất lượng gần như nguyên bản
- Cho phép training trên GPU 24GB VRAM

3.4. Cấu Hình Training

Thành phần	Giá trị	Mục đích
Max sequence length	5000 tokens	Đủ cho prompt dài với few-shot examples
Batch size	4 per device	Cân bằng memory và tốc độ
Gradient accumulation	8 steps	Effective batch size = 32
Learning rate	5e-5	Tối ưu cho fine-tuning
Weight decay	0.01	Regularization
Epochs	1	Tránh overfitting

Kỹ thuật đặc biệt:

- **train_on_responses_only**: Chỉ tính loss trên câu trả lời, bỏ qua phần prompt
- **Gradient checkpointing**: Giảm 40% memory footprint
- **Mixed precision (bfloat16)**: Tăng tốc độ training

3.5. Inference với vLLM

vLLM Engine:

- PagedAttention: Quản lý KV cache hiệu quả
- Continuous batching: Xử lý đồng thời nhiều requests
- Tốc độ nhanh hơn HuggingFace Transformers 5-10 lần

Sampling Parameters:

- Temperature: 0.1 (giảm randomness, tăng consistency)
- Top-p: 0.8 (nucleus sampling)
- Top-k: 5 (giới hạn tokens ưu tiên)
- Max tokens: 64 (đủ cho output ngắn gọn)

4. QUY TRÌNH XỬ LÝ

4.1. Training Pipeline

1. **Load Base Model**: Qwen3-4B với 4-bit quantization
2. **Apply LoRA Adapters**: Gắn adapters vào attention và FFN layers
3. **Prepare Dataset**:
    - Load training data và few-shot examples
    - Sample 5 few-shots cho mỗi mẫu (deterministic theo ID)
    - Build prompts với instruction + examples + query
    - Apply chat template chuẩn của Qwen3
4. **Training**: SFTTrainer với train_on_responses_only
5. **Save Models**:
    - LoRA adapter weights (nhỏ gọn)
    - Merged 16-bit model cho vLLM inference

4.2. Inference Pipeline

1. **Load Model**: Merged 16-bit model vào vLLM engine
2. **Prepare Input**:
    - Đọc test data
    - Sample few-shots cho từng mẫu
    - Build prompts và apply chat template
3. **Generate**: Batch generation với vLLM (batch size = 8)
4. **Parse Output**: Extract label theo thứ tự ưu tiên (extrinsic → intrinsic → no)
5. **Save Results**: Xuất predictions ra CSV

5. Lưu Ý

⚠ **Về tính nhất quán**: Do sử dụng LLM với sampling (temperature > 0), kết quả có thể có chênh lệch giữa các lần chạy. Đây là đặc tính của generative models.

6. KẾT LUẬN

Hệ thống kết hợp thành công nhiều kỹ thuật hiện đại trong NLP:

Điểm mạnh:

- Few-shot learning cung cấp context học tập trực quan cho model
- LoRA fine-tuning tối ưu hiệu quả với tài nguyên hạn chế
- vLLM đảm bảo tốc độ inference cao
- Prompt engineering định hướng model chính xác

Kỹ thuật nổi bật:

- Kết hợp in-context learning và parameter-efficient fine-tuning
- Quantization và LoRA giảm đáng kể tài nguyên cần thiết
- Deterministic sampling đảm bảo reproducibility
- Custom training strategy (train_on_responses_only)

Kết quả: Một pipeline hiệu quả có khả năng phát hiện hallucination trong mô hình ngôn ngữ tiếng Việt với độ chính xác cao, tốc độ xử lý nhanh và yêu cầu tài nguyên hợp lý.

PHỤ LỤC: CÔNG CỤ VÀ THƯ VIỆN

- **Unsloth**: Fast LoRA training & quantization framework
- **vLLM**: High-performance inference engine với PagedAttention
- **TRL (Transformer Reinforcement Learning)**: SFTTrainer framework
- **HuggingFace Transformers**: Model loading và tokenization
- **PyTorch**: Deep learning backend

Yêu cầu phần cứng:

- GPU: NVIDIA RTX 4090 (24GB VRAM)
- RAM: 32GB
- Storage: 512GB
